Pepadun v1i1 2020 125-132

by Yunda heningtyas

Submission date: 17-Feb-2022 09:46AM (UTC-0500)
Submission ID: 1756580132
File name: Jurnal_-_Pepadun_v1i1_2020_125-132.pdf (553.72K)
Word count: 2213
Character count: 14016

Vol 1 No. 1, November 2020, pp. 125-132 **Jurnal Pepadun**

PENGEMBANGAN SISTEM GRADER DI JURUSAN ILMU KOMPUTER FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM UNILA

[1]Mei Rusfandi, [2]Didik Kurniawan, dan [3]Ardiansyah

[123] Jurusan Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampun
e-mail : [1]meirusfandi100@gmail.com, [2]didikunila@gmail.com, [3]ardiansyah@fmipa.unila.ac.id

Abstract *— Grader is a system to complement the demand of online learning system at the Computer Science Department, Faculty of Mathematics and Natural Science, University of Lampung. The existed virtual class system facilitates assessments with automatic correction for assignments and tests. On the other hand, a grader aims to generate automatic correction for programming tasks. This research implements extreme programming method to build a grader system. The system was built using the CodeIgniter PHP framework. Developments carried out in this study include the addition of user categories to admin, editor and participants; as well as the addition of the share code feature. Black box testing was conducted twice to validate the performance of the system, in which fulfilled all the requirements and set to release.*

Keywords: *automation testing; black box testing; extreme programming; grader system*

1. PENDAHULUAN

Virtual class merupakan pembelajaran yang memuat konten-konten digital yang dapat diakses dan dipertukarkan dimana saja, dari mana saja, dan kapan saja. *Virtual class* dapat memantau kemajuan proses belajar mengajar dosen dan mahasiswa serta dapat digunakan untuk proses pendidikan jarak jauh [1]. *Virtual class* harus mampu menciptakan pembelajaran yang kondusif, interaktif dan dinamis dengan menerapkan tujuan pembelajaran yang jelas dan spesifik, menyusun bahan ajar yang baik, dan fasilitas komunikasi timbal balik antara dosen dan mahasiswa. *Virtual class* harus menyediakan fasilitas perkuliahan yang terintegrasi (tugas, bahan kuliah, rencana pembelajaran, dan penilaian hasil belajar) dan dapat mengukur pencapaian kompetensi mahasiswa. *Virtual class* juga dirancang agar mahasiswa dapat berbagi (*share*) hasil karya dan bertukar pengalaman dalam menerapkan pengetahuan yang telah diperoleh melalui *video conference* atau simulasi secara *online* [2].

Media pembelajaran berbasis *virtual class* dengan bantuan Google Drive direkomendasikan untuk digunakan pada proses pembelajaran di perkuliahan. Hasil ini didapatkan dari perhitungan validitas yang didapatkan berdasarkan aspek isi dan tampilan media yang sudah dikembangkan. Media pembelajaran ini diimplementasikan pada mata kuliah termodinamika pada program studi fisika FKIP Universitas Pasir Pengaraian [3].

Virtual class berbasis *website* telah tersedia di Universitas Lampung, menggunakan Moodle dan Wordpress yang memenuhi kriteria media pembelajaran yang efektif, efisien, praktis, dan interaktif. Bagi dosen, melalui sistem ini mampu dilakukan pengaplikasian pembelajaran dan evaluasi hasil belajar mahasiswa [4]. Akan tetapi, *virtual class* yang telah diimplementasikan masih belum mampu menangani tugas mahasiswa yang berupa tugas pemrograman, serta manajemen nilai masih dilakukan secara manual. Tugas yang bersifat pemrograman masih dikoreksi dan dinilai secara manual. Selain permasalahan tersebut, tidak sedikit mahasiswa yang masih belum dapat membuat program sendiri atau kurang percaya diri dengan kemampuannya. Oleh karena itu, dibutuhkan suatu *virtual class* yang mampu menangani tugas pemrograman, memanajemen nilai pemrograman, membantu mahasiswa berlatih *coding*, dan bahkan dapat digunakan untuk lomba pemrograman pada acara Pekan Raya Jurusan (PRJ) Ilmu Komputer. Universitas Lampung.

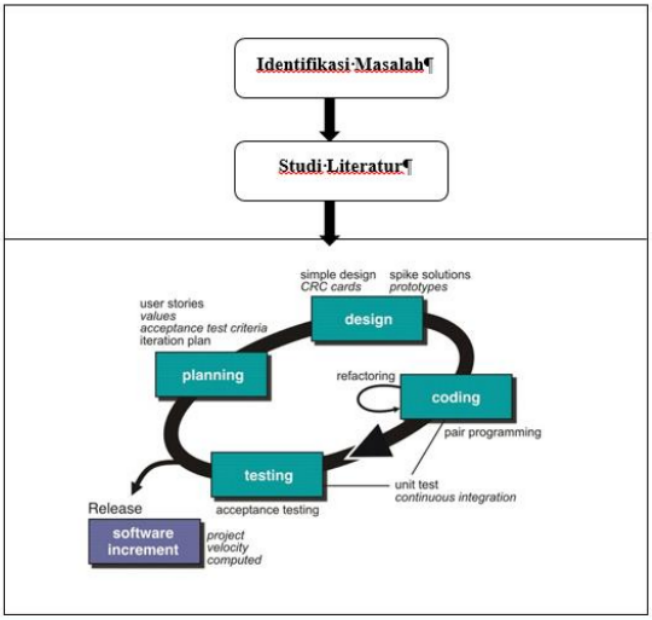
Grader atau sistem *grading* merupakan sistem yang digunakan untuk melakukan pengujian algoritme pemrograman dalam Bahasa Pemrograman Pascal, Java, C/C++, dan sebagainya. *Grader* digunakan oleh Tim Olimpiade Komputer Indonesia (TOKI) untuk melatih siswa-siswi SMA / sederajat untuk dapat mengikuti perlombaan Olimpiade Sains Nasional (OSN) bidang Komputer. *Grader* dibangun mengikuti aturan *International Olympiad in Informatics* (IOI) dan *Association for Computing Machinery – International Collegiate Programming Contest* (ACM-ICPC) [5].

MOE Contest Environment adalah sebuah sistem untuk menyelenggarakan kompetisi pemrograman yang serupa dengan semangat pada *International Olympiad in Informatics* (IOI). Peserta menyelesaikan tugas pemrograman, mengumpulkan kode program (*source code*) solusi mahasiswa, yang kemudian diuji secara otomatis pada satu set tes masukan [6].

Berdasarkan permasalahan yang dipaparkan, diperlukan batasan dalam menjalankan penelitian. Adapun batasan masalah pada penelitian ini adalah sistem dikembangkan untuk melakukan *auto correction* / *auto testing* untuk semua bahasa pemrograman, dan dikembangkan dengan *framework* Codeigniter.

2. METODOLOGI PENELITIAN

Tahapan penelitian merupakan acuan dalam melakukan penelitian sehingga dapat dilakukan sesuai rancangan yang telah ditetapkan. Alur penelitian ini melalui tiga tahapan, yaitu identifikasi masalah, studi literatur, dan pengembangan yang diilustrasikan pada Gambar 1.



Gambar 1. Alur penelitian

2.1. Identifikasi Masalah

Identifikasi masalah merupakan tahapan untuk menganalisis permasalahan-permasalahan yang ada, baik proses pengembangan, perbaikan dan penambahan fitur, serta proses bisnis yang telah ada. Terdapat dua

kebutuhan yang perlu dikembangkan dari sistem *grader*, yaitu kebutuhan fungsional dan kebutuhan non-fungsional.

Kebutuhan fungsional dari sistem *grader* adalah sebagai berikut.

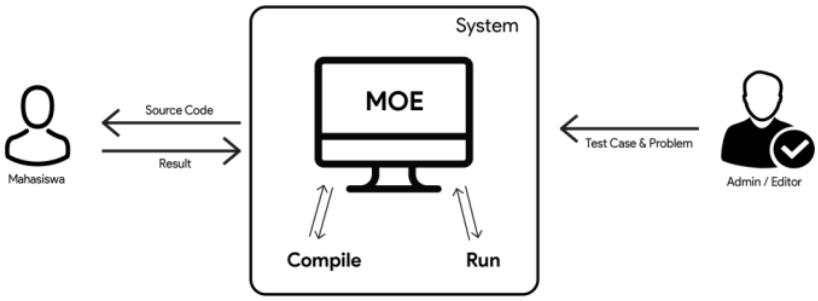
1. Membuat *contest* / kelas
2. Membuat soal / *problem* yang disertai dengan *testcase*
3. Menambah *user*
4. Melakukan *compile* dan *running* program secara otomatis
5. Menambah *submission* jawaban
6. Membuat *clarification*
7. Menambah bahasa pemrograman

Kebutuhan non-fungsional dari sistem *grader* adalah sebagai beirkut.

1. Keamanan akses pengguna (*Username* dan *Password*)
2. *User interface* yang mudah dipahami
3. Proses *regrade*

2.2. Studi Literatur

Tahap studi literatur melakukan proses pencarian informasi yang dibutuhkan untuk sistem, dimulai dari awal proses hingga akhir, sehingga dapat diketahui apa saja kebutuhan sistem. Hasil yang diperoleh dari tahap ini berupa informasi berupa teori-teori dan hasil penelitian yang sudah dilakukan mengenai permasalahan yang akan diteliti. Model pengembangan sistem *grader* tersedia pada Gambar 2.



Gambar 2. Model *Grading System Development*

Pada Gambar 2, diperoleh informasi bahwa proses *compile* dan *run source code* hasil *submission* dari mahasiswa / *contestant* dilakukan oleh *MOE Contest Environment* sebagai proses yang berjalan di dalam sistem. Pengembangan sistem dilakukan dengan mengikuti alur pada Gambar 2 tersebut. Dosen atau *editor* bertugas untuk mengisi *testcase* dan *problem* dari tiap kelas atau *contest* yang telah dibuat.

2.3. Metode

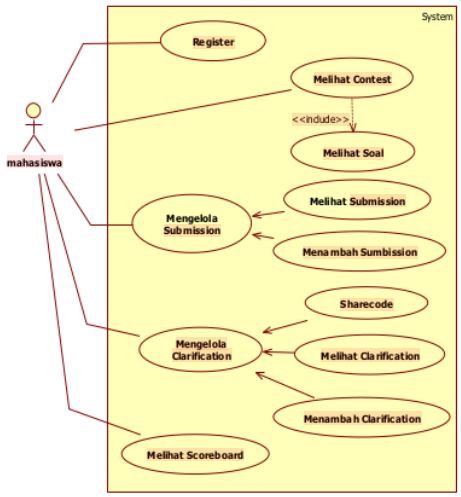
Metode *Extreme Programming* merupakan metode pengembangan perangkat lunak yang menggunakan beberapa tahapan dalam prosesnya. Pengembangan perangkat lunak menggunakan metode *extreme programming* yaitu *planning, design, coding*, dan *testing*. Proses dari metode *extreme programming* tersedia sebagai bagian dari Gambar 1 sebelumnya.

3. HASIL DAN PEMBAHASAN

Sistem *grader* merupakan pengembangan dari sistem yang sudah ada dengan fungsi yang sama dan dengan penambahan fitur berupa kategori pengguna, *export* nilai ke dalam *file excel*, dan fitur *share code*. Tahapan penelitian ini mengikuti tahapan yang ditampilkan dalam bentuk diagram alir pada Gambar 1.

3.1. *Planning*

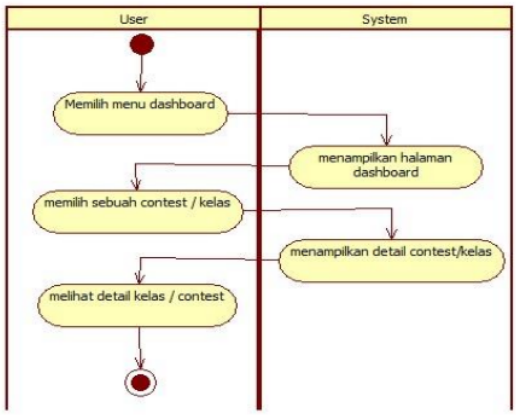
Proses *planning* menggunakan diagram *use case* untuk menggambarkan hubungan antara aktor dengan sistem. Pada sistem ini, terdapat tiga kategori pengguna, yaitu *contestant/user/* mahasiswa, *editor* / dosen / asisten dosen, dan *admin*. *Use case diagram* untuk *user* / mahasiswa / *contestant* dapat dilihat pada Gambar 3.



Gambar 3. *Use case diagram* mahasiswa

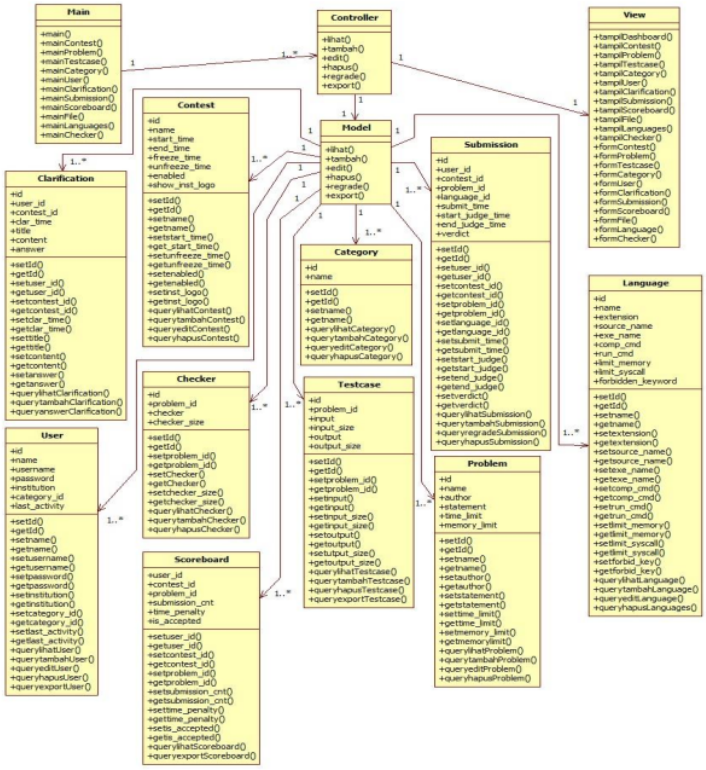
3.2. *Design*

Tahap perancangan berfungsi untuk memberikan gambaran prototipe sistem. Pada tahap desain sistem, dilakukan perancangan diagram aktivitas (*activity diagram*), kelas diagram (*class diagram*), dan diagram sekuens (*sequence diagram*). *Activity diagram* digunakan untuk menggambarkan serangkaian aliran dari aktivitas. *Activity diagram* juga berguna untuk mendeskripsikan aktivitas-aktivitas yang dibentuk dalam satu operasi, sehingga dapat melakukan aktivitas lainnya. Gambar 4 menunjukkan *activity diagram* untuk melihat *dashboard* mahasiswa.



Gambar 4. *Activity diagram* untuk melihat *dashboard* mahasiswa

Pada *class diagram* terdapat nama-nama tabel dan atribut yang digunakan pada sistem *grader*. *Class diagram* pada sistem ini menggunakan pendekatan *Model, View, and Controller* (MVC) dan diilustrasikan pada Gambar 5.

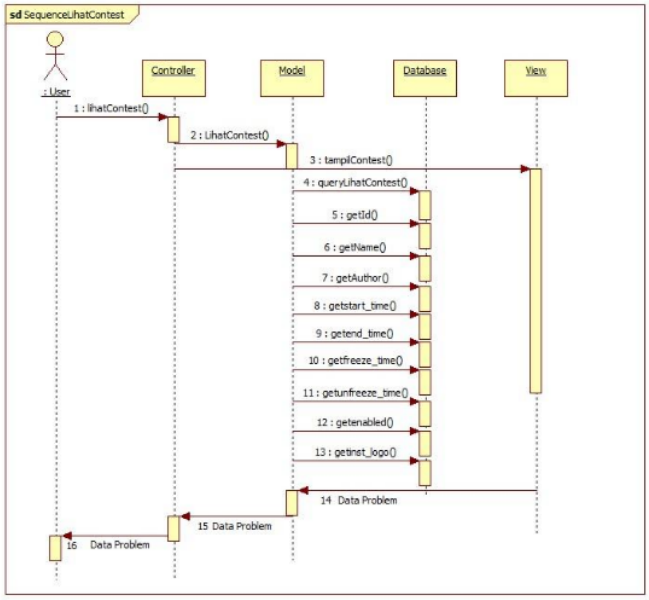


Gambar 5. *Class diagram*

Sequence diagram menggambarkan interaksi antara objek dan sistem, berupa pesan terhadap waktu dan merupakan hubungan antara sistem dengan pengguna. *Sequence diagram* digambarkan sesuai dengan cara sistem memproses permintaan dari pengguna. Gambar 6 menunjukkan *sequence diagram* pada sistem *grader* untuk melihat *contest*.

Sequence diagram melihat *contest* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

1. Pengguna memilih menu lihat *contest* pada *controller contest*.
2. *Controller contest* mengakses *model contest* untuk mengambil data dari *database contest*.
3. *Controller* menampilkan data *contest* ke *view contest*.



Gambar 6. *Sequence Diagram* untuk melihat *contest*

3.3. *Coding*

Tahap *coding system* dilakukan dengan mengimplementasikan permasalahan yang sudah didapatkan berdasarkan desain yang dibuat. Proses *coding system* menggunakan bahasa pemrograman PHP dengan menggunakan *framework* CodeIgniter dan MySQL *database*. Proses *coding* menggunakan kode program yang sudah tersedia, dan mengembangkan kode program tersebut agar sesuai dengan *requirement* yang sudah dianalisis sebelumnya. Tabel 1 menunjukkan algoritme fitur *share code*.

Tabel 1. Algoritme fitur *share code*

No	Algoritme fitur *share code*
1	*Submission*
2	*Show submission*
3	*Click sharecode*
4	*Get link url*
5	*Share code to another user*

Tabel 1 menjelaskan bagaimana fitur *share code* bekerja. Pertama, pengguna akan memilih *submission*, lalu sistem akan menampilkan *submission* yang dipilih. Selanjutnya, pengguna memilih dan menekan tombol *share code*, lalu sistem akan membuka *tab* baru dengan memberikan *link url* yang dapat dibagikan ke pengguna lain. Tabel 2 menunjukkan algoritme *automation testing*.

Tabel 2. Algoritme *Automation Testing*

No	Algoritme *automation testing*
1	Start
2	Run db
3	Prev_verdict = 0
4	If sub[‘verdict’] < 0
5	Prev_verdict = -sub[‘verdict’]
6	If prev_verdict == 99
7	Set suvmission verdict
8	Else
9	Update db
10	Cek compile_status
11	If compile_status == 0
12	Verdict = run submission
13	Else
14	Verdict = 1
15	Set submission verdict
16	Finish

Berikut ini merupakan penjelasan proses dari algoritme pada Tabel 2.

1. Proses dimulai dengan pengguna mengirimkan *submission* dari sebuah *problem*.
2. Sistem mengambil data dari *database* jika pengguna sudah pernah melakukan *submission* untuk *problem* yang dipilih,
3. Buat nilai prev_verdict dengan nilai awal 0,
4. Cek dari *database*, jika *submission verdict* nilainya kurang dari 0, maka atur nilai prev_verdict menjadi nilai dari *submission verdict*.
5. Cek nilai prev_verdict, jika nilainya sama dengan 99, maka atur nilai *submission verdict*.
6. Jika nilai prev_verdict tidak sama dengan 99, maka cek *compile status*.
7. Jika *compile status* bernilai 0, atau tidak bermasalah, maka jalankan *submission*.
8. Jika nilai *compile status* tidak sama dengan 0, maka atur nilai *verdict* dengan 1.
9. Jika sudah melakukan *run submission*, maka atur nilai *submission verdict* pada *database*.
10. *Finish*.

3.4. *Testing*

Proses pengujian dilakukan menggunakan metode *black box testing*. *Black box testing* merupakan tahap pengujian yang memperhatikan fungsionalitas dari sistem. Proses ini dilakukan untuk memastikan sistem yang dibuat telah bekerja sesuai dengan ketentuan yang telah ditetapkan. Pada pengujian ini, dilakukan dengan membagi *domain* masukan ke beberapa *testcase* yang dibuat.

Pengujian pertama dilakukan dan memberikan hasil bahwa *form* untuk menambah *submission* kurang sesuai, dan informasi yang berikan pada saat melihat *detail submission* kurang lengkap. Berdasarkan hasil ini, dilakukan *cross check* pada tahap *planning* dan *design*. Pada kedua tahap ini tidak terdapat masalah, sehingga dilanjutkan ke tahap *coding* program.

Pengujian dilakukan dengan memberikan kuisioner kepada responden. Kuisioner tersebut berisi beberapa pertanyaan terkait sistem kepada 3 pengguna, yaitu *admin, editor* dan *contestant*. Pengguna memberikan respon yang terjadi saat menggunakan sistem dengan menjawab pertanyaan, dan memberikan kritik maupun saran yang dapat membangun perkembangan sistem.

Berdasarkan pengujian yang telah dilakukan, terdapat beberapa masukan yang diberikan oleh rersponden penguji. Adapun beberapa masukan tersebut adalah sebagai berikut.

1. Bahasa pemrograman yang ditampilkan tidak semua bahasa pemrograman
2. Penambahan fitur *solved* pada halaman *clarification*
3. *Clarification* dapat dimaksimalkan untuk semua *contestant*
4. Perbaikan untuk koreksi bahasa pemrograman Python
5. Perbaikan pada halaman *register*
6. Penambahan fitur lupa *password*

4. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut.

1. Sistem *grader* telah berhasil dikembangkan dan diimplementasikan ke *server* laboratorium Jurusan Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampung dan dapat diakses pada laman www.lab.ilkom.unila.ac.id/grader.
2. Halaman *register* mahasiswa berjalan dengan baik.
3. Fitur *share code* berhasil diimplementasikan dan menghasilkan sebuah *link* baru yang dapat diakses tanpa melakukan *login*.
4. Sistem dikembangkan dengan 3 kategori pengguna, yaitu *admin, editor* dan *contestant* (mahasiswa).
5. Sistem dapat digunakan untuk semua bahasa pemrograman.
6. *Admin* dan *editor* dapat mengunduh *scoreboard* ke dalam *file excel* berekstensi .xls.
7. Hasil pengujian yang diujikan kepada tiga kategori pengguna berhasil dilakukan.

DAFTAR PUSTAKA

[1] Soelistijadi, R., Priambodo, A., dan Amin, F. 2012. "Aplikasi E-Commerce Sentra Batik di Kota Semarang Sebagai Salah Satu Upaya Media Promosi dan Transaksi Secara Online," *Jurnal Teknologi Informasi Dinamik*, vol. 17, no. 1, pp. 67-74.

[2] Prassida, G. F. dan Muklason, A. 2011. "Virtual Class Sebagai Strategi Pembelajaran Untuk Peningkatan Kualitas Student Centered Learning Di Perguruan Tinggi," *Jurnal Institut Teknologi Sepuluh November*, vol. 1, no. 2, pp. 95-98.

[3] Rhomdani, R. W. 2016. "Pengembangan Virtual Class Matematika Berbasis Web Menggunakan Moodle dan Wordpress di Universitas Muhammadiyah Jember," *Jurnal Universitas Muhammadiyah Jember*, vol. 1, no. 1, pp. 19-31.

[4] Sohibun, S. dan Ade, F. Y. 2017. "Pengembangan Media Pembelajaran Berbasis Virtual Class Berbantuan Google Drive," *Jurnal Keguruan dan Ilmu Tarbiyah*, vol. 2, pp. 121-129.

[5] MOE Contest Environment. 2015. The Moe Contest Environment. [Online]. https://www.ucw.cz/moe/

[6] Regrader. 2015. Fushar Regrader. [Online]. https://github.com/fushar/regrader

Pepadun v1i1 2020 125-132

ORIGINALITY REPORT

99%	99%	10%	13%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	pepadun.fmipa.unila.ac.id Internet Source	99%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		